Reconstructing Breakage Fusion Bridge Architectures Using Noisy Copy Numbers

Shay Zakov* and Vineet Bafna

Department of Computer Science and Engineering,
University of California, San Diego, CA, USA
szakov@eng.ucsd.edu

Abstract. The *Breakage Fusion Bridge* (*BFB*) process is a key marker for genomic instability, producing highly rearranged genomes in relatively small number of cell cycles. While the process itself was observed during the late 1930's, little is known about the extent of BFB in tumor genome evolution. This is partly due to methodological requiring the rare observation of a spontaneous BFB occurence, or rigorous assays for identifying BFB-modified genomes after the process has ceased. Moreover, BFB can dramatically increase copy numbers of chromosomal segments, which in turn hardens the tasks of both reference assisted and *ab initio* genome assembly.

Based on available data such as *Next Generation Sequencing* (NGS) and *Array Comparative Genomic Hybridization* (aCGH) data, we show here how BFB evidence may be identified, and how to predict all possible evolutions of the process with respect to observed data. Specifically, we describe practical algorithms that, given a chromosomal arm segmentation and noisy segment copy number estimates, produce all segment count vectors supported by the data that can be produced by BFB, and all corresponding BFB architectures. This extends the scope of analyses described in our previous work, which produced a single count vector and architecture per instance.

We apply these analyses to a comprehensive human cancer dataset, demonstrate the effectiveness and efficiency of the computation, and suggest methods for further assertions of candidate BFB samples. An online Appendix, the source code of our tool, and analyses results, are available at `http://cseweb.ucsd.edu/~vbafna/bfb`.

1 Introduction

The origin of a tumor cell is marked by genomic instability [9]. Spontaneous, viral, or other kinds of mechanisms may cause genomic segment deletions, duplications, translocations, inversions, etc., producing rearranged genomes with a possibly malignant nature. Thus, decoding mechanisms that generate rearranged genomes is critical to understanding cancer. Numerous mechanisms were proposed, including the faulty repair of double-stranded DNA breaks by recombination or end-joining and polymerase hopping caused by replication fork

* Corresponding author.

© Springer International Publishing Switzerland 2014

collapse [5,10]. These mechanisms are generally not directly observable, so their elucidation requires the deciphering of often subtle clues after genomic instability has ceased. An important source of information in this respect is the *architecture* of the rearranged genome, i.e. the description of its chromosomes in terms of concatenations of segments from the original genome.

Breakage Fusion Bridge (BFB) is one model of a genome rearrangement process, which was first proposed by Barbara McClintock in the 1930's [13,14]. Recently, it has seen renewed interest as a possible mechanisms in tumor genome evolution [3,4]. BFB begins with a telomeric loss on a chromosome, including a loss of a sequential pattern that signals the location of chromosome termination. During cell division the telomere-lacking chromosome replicates, and its two sister chromatids fuse together (possibly due to some DNA repair mechanism falsely induced by the cell). This fusion produces a dicentric chromosome of palindromic structure, which is later torn apart at some random point as the centromeres of the dicentric chromosome migrate to opposite poles of the cell. One part of the torn chromosome includes the fusion region and some tandemly inverted chromosomal suffix duplication, and the other part lacks the corresponding suffix. The two daughter cells receive these rearranged chromosomes, both are missing the telomeric region, and the cycle can repeat again (Fig. 1).

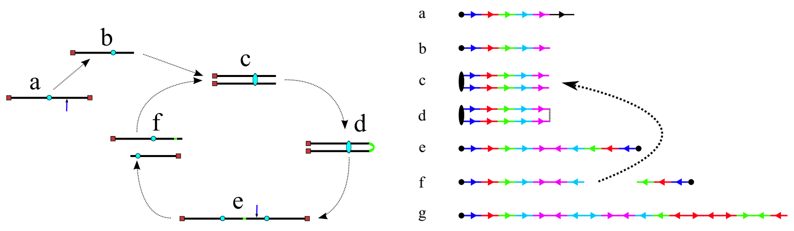


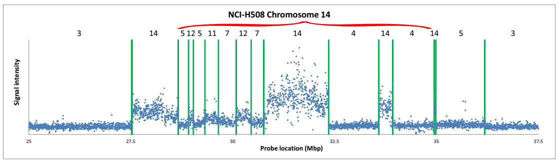
Fig. 1. The BFB process. (a) A normal chromosome. (b) The chromosome looses its telomere. (c) The chromosome is duplicated during cell devision. (d) Sister chromatids are fused together. (e) Centromeres migrate to opposite poles of the cell. (f) The fused chromosome is torn apart at some random position, causing one copy to have an inverted suffix duplication, while the other copy has a trimmed suffix. Both copies lack a telomere, and therefore may undergo additional BFB cycles. (g) After several BFB cycles, the chromosome architecture exhibits significant increases in segment copy numbers, as well as fold-back patterns.

In contrast to other mechanisms, BFB can actually be observed in progress using methods that have been available for decades [14]. Cytogenetic techniques can reveal the anaphase bridges, dicentric chromosomes, and homogeneously staining regions that have long been the canonical evidence for BFB. However, these techniques are useful only in cases where the BFB cycles are ongoing. While useful in understanding the mechanism, they do not address the question of whether BFB occurs extensively in evolving tumor genomes.

Recently, researchers (including us) have started looking at modern available data in order to demonstrate BFB occurrence after the process has ceased, including *Fluorescent In Situ Hybridization* (FISH), *Array Comparative Genomic Hybridization* (aCGH), and *Next Generation Sequencing* (NGS) data. These methods take advantage of distinctive BFB features exposed by such data, including the abundance of fold-back inversions (i.e. duplicated chromosomal segments arranged in a head-to-head orientation) [3,4], patterns of interleaving segments of alternating orientations [12,17], and combinatorial properties of segment counts when copy number variations are due to BFB [11,20]. In fact, if the architecture of the rearranged genome is known, it is possible to decide if this architecture can be produced by BFB [11].

Partial knowledge regarding the architecture can be reveled by FISH analyses [12], which uses fluorescence markers to identify the physical locations of predetermined sequences on the rearranged genome. However, such experiments are relatively expensive, and can only be performed in a small number of cases. A more common measurement is NGS data, which contain a big set of short sequenced reads extracted from a donor genome. Such data is typically used for predicting the entire donor genomic sequence by computationally assembling the reads, sometimes facilitated by consulting a similar pre-sequenced reference genome. Unfortunately, BFB and other mechanisms can produce massively rearranged and highly repetitive genomes. This hardens the task of assembly-based sequencing due to the multiple ambiguous manners the repetitive reads may be assembled, and the lack of a relevant reference template. Nevertheless, NGS data can still be analyzed in order to infer some indirect information regarding the donor genome architecture [1,6,15,19]. After aligning the reads against a reference genome, their genomic location distribution can be used in order to identify segments on the reference genome of coherent read coverage, and to estimate the number of times each such segment repeats in the donor genome. We will refer to the output of the latter kind of analysis as *copy number data*. Other methods to obtain copy number data are based on analyzing aCGH data [7,8,16,18] (Fig. 2). Due to the noisy nature of both NGS and aCGH data, count estimates may be inaccurate, and the true segment count is likely to fall within some interval of integers around the estimated value. We use the term *noisy copy number data* when referring to information regarding such intervals of possible count values. In addition to copy number data, NGS data can be used in order to produce contigs (chromosomal segments which may be assembled unambiguously), and aberrant segment adjacencies can be exposed by discordant reads, restricting the set of possible contig-based architectures.

In previous work [11,20], we showed how to analyze noisy copy number data in order to decide is it likely to observe the input data under the assumption the underlying rearrangement process is BFB. Specifically, we designed algorithms that produce a single BFB architecture over the given segments in which segment counts are supported by the data, if such an architecture exists. We applied these algorithm in order to analyze a comprehensive aCGH dataset of cancer cell lines [2], as well as sequence data from primary tumors [4], and



(a) aCGH data

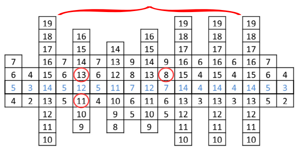

(b) Noisy copy number data

Fig. 2. (a) aCGH data for a part of the q-arm of human chromosome 14 in the NCI-H508 cell line. Each data point corresponds to a probe on the array, where its x-coordinate gives the probe's sequence chromosomal position, and y-coordinate gives its measured intensity (log-ratio). The data points are clustered into segments, and an estimated segment copy number appears above each segment. (b) Possible count intervals around the estimated counts. The counts in the region under the red curly bracket are supported by a BFB architecture, if changing the count estimate of the second segment in this region from 12 to 13 or to 11, and of the seventh segment from 7 to 8. Data is taken from [2] (segmentation and copy number analysis were computed using the PICNIC software [8]).

identified a small subset of candidate samples exhibiting BFB hallmarks. Here, we extend the scope of the analysis, and describe algorithms that report *all* count settings supported by the data which can be explained by BFB, and all corresponding BFB architectures. Although the theoretical time bounds for these new algorithms may be exponential, we show that in practice they are efficient, and apply an Informed Search optimization that further improves their practical efficiency.

Our proposed algorithms satisfy an important need, therefore. While our work postulates the existence of BFB using statistical arguments, additional physical assertions can be obtained with FISH and aberrant read analyses. Starting with noisy copy number data, our tool can be used to enumerate all possible BFB architectures. These candidate architectures can then be used towards a small set of FISH experiments (with a limited number of fluorescence markers) to validate and refine the genomic architecture.

2 Formalism and Previous Results

Computational BFB-related problems were previously formulated in [11,20]. For completeness, we give here the main definitions from these works.

A *DNA segment* σ is a string over the DNA nucleotide alphabet A, C, G, T. The *reversed segment* of a segment σ, denoted here by $\overline{\sigma}$, is the string obtained by reading σ backwards, and replacing each nucleotide with its complementary nucleotide ($A \leftrightarrow T, C \leftrightarrow G$). For example, the reverse of a segment $\sigma = CGGAT$ is the segment $\overline{\sigma} = ATCCG$. In the rest of this paper, it is assumed we operate on a given chromosomal arm with a fixed segmentation, and denote its list of k segments by $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$, ordered from the centromeric segment σ_1 to the telomeric segment σ_k. The term "string" refers to a genomic architecture over these segments, i.e. a concatenation of segments from Σ and their reversed forms. Greek letters $\alpha, \beta, \gamma, \rho$ denote strings, and bar notation indicates reversed strings. For example, if $\alpha = \sigma_1\sigma_3\overline{\sigma}_2$, $\overline{\alpha} = \sigma_2\overline{\sigma}_3\overline{\sigma}_1$. An empty string is denoted by ε. The notation $\alpha_{l,t}$ represents the string $\sigma_l\sigma_{l+1}\ldots\sigma_t$ (thus when $t < l$, $\alpha_{l,t} = \varepsilon$). To facilitate reading, $\sigma_1, \sigma_2, \sigma_3, \ldots$ are replaced by A, B, C, ... in concrete examples.

A *BFB cycle* applied over a chromosomal arm can be viewed as a special rearrangement procedure, in which some telomeric suffix of the arm is duplicated, inverted, and concatenated tandemly at the telomeric end of the arm. A *BFB process* is a consecutive application of BFB cycles. This notion is formally captured by the following definition.

Definition 1. *For two strings α, β, say that $\alpha \xrightarrow{BFB} \beta$ if $\alpha = \beta$, or $\alpha = \rho\gamma$ for some strings ρ, γ such that $\gamma \neq \varepsilon$, and $\rho\gamma\overline{\gamma} \xrightarrow{BFB} \beta$. Say that α is an l-*BFB string *if $\alpha_{l,t} \xrightarrow{BFB} \alpha$ for some t, and say that α is a* BFB string *if it is an l-BFB string for some l.*

Note that by definition $\varepsilon = \alpha_{l,l-1}$ is an l-BFB string for every $l \geq 1$. The *count vector* $\vec{n}(\alpha) = [n_1, n_2, \ldots, n_k]$ of a string α is a vector of integers, where for every $1 \leq l \leq k$, n_l is the total number of occurrences of σ_l and $\overline{\sigma}_l$ in α. For example, for $\alpha = \text{ABCD}\bar{\text{D}}\bar{\text{C}}\text{C}$, $\vec{n}(\alpha) = [1, 1, 3, 2]$. Say that a vector $\vec{n}$ is a *BFB vector* if there exists some BFB string α such that $\vec{n} = \vec{n}(\alpha)$. In the previous example $\vec{n}(\alpha)$ is a BFB vector, due to the BFB process $\alpha_{1,4} = \text{ABCD} \xrightarrow{\text{BFB}} \text{ABCD}\bar{\mathbf{D}}\bar{\mathbf{C}} \xrightarrow{\text{BFB}} \text{ABCD}\bar{\text{D}}\bar{\text{C}}\mathbf{C} = \alpha$.

The computational analyses presented in this paper aim to detect evidence for BFB, given a pre-analyzed segmentation of the genome and corresponding copy number data. We assume that noisy copy number data is represented by a *weight function* $W = \{w_{l,n} \mid 1 \leq l \leq k, n = 0, 1, 2, \ldots\}$, where $w_{l,n}$ is a nonnegative *weight* of the count n with respect to the l-th segment. It may be assumed w.l.o.g. that all weights $w_{l,n}$ satisfy $0 \leq w_{l,n} \leq 1$. The *weight* of a count vector $\vec{n} = [n_1, n_2, \ldots, n_k]$ is given by $W(\vec{n}) = \prod\limits_{1 \leq i \leq k} w_{i,n_i}$, and by assumption $0 \leq W(\vec{n}) \leq 1$. In some cases, we refer to prefixes $\vec{n}_{1,l-1} = [n_1, n_2, \ldots, n_{l-1}]$ and suffixes $\vec{n}_{l,k} = [n_l, n_{l+1}, \ldots, n_k]$ of $\vec{n}$, which may be empty if $l = 1$ or $l = k+1$, respectively. Define the weights of such sub-vectors accordingly, i.e. $W(\vec{n}_{1,l-1}) = \prod\limits_{1 \leq i < l} w_{i,n_i}$ and $W(\vec{n}_{l,k}) = \prod\limits_{l \leq i \leq k} w_{i,n_i}$, where the weight of an empty vector is 1 by definition. Thus, for every $1 \leq l \leq k+1$, $W(\vec{n}) = W(\vec{n}_{1,l-1}) \cdot W(\vec{n}_{l,k})$.

If some data analysis produces segment count probabilities $\Pr(n_l = n)$ for every segment σ_l and every count $n = 0, 1, 2, \ldots$, weights can be set to these probabilities choosing $w_{l,n} = \Pr(n_l = n)$. This way, the weight of a count vector is the probability this vector reflects the true segment counts, given the observed data. Another way to set weights given such probabilities would be to choose weights by setting $w_{l,n} = \frac{\Pr(n_l = n)}{\Pr\left(n_l = n_l^*\right)}$, where n_l^* is the most likely count for the l-th segment. Here, the weight of a count vector gives the ratio between its probability and the probability of a most likely vector. Nevertheless weights are more general than probabilities, and can be used as a heuristic count error modeling even when no probabilistic model is available.

In [20], several variants of BFB problems where formulated. Below we restate these problems, and add two new variants addressed in the current work:

BFB Problem Variants
Input: *a count vector $\vec{n}$, or a weight function W and a weight $0 < \eta \leq 1$.*

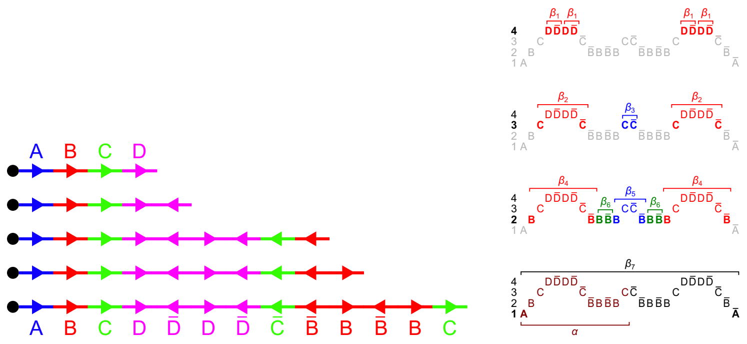
1. **The decision variant** [20]: *given $\vec{n}$, decide if $\vec{n}$ is a BFB vector.*
2. **The string search variant** [20]: *if $\vec{n}$ is a BFB vector, find a BFB string α such that $\vec{n} = \vec{n}(\alpha)$.*
3. **The vector search variant** (or **the distance variant** in [20]): *given W and η, report a maximum weight BFB vector $\vec{n}$ in case there exists such a vector with $W(\vec{n}) \geq \eta$, and otherwise report "FAILED".*
4. **The exhaustive vector search variant:** *given W and η, report all BFB vectors $\vec{n}$ with $W(\vec{n}) \geq \eta$.*
5. **The exhaustive string search variant:** *given W and η, report all BFB strings α such that $W(\vec{n}(\alpha)) \geq \eta$.*

For a count vector $\vec{n}$, define $N(\vec{n}) = \sum_{1 \leq l \leq k} n_l$ and $\tilde{N}(\vec{n}) = \sum_{1 \leq l \leq k} \log(n_l)$. Note that $N(\vec{n})$ is the total length of a string admitting $\vec{n}$, and $\tilde{N}(\vec{n})$ is proportional to the number of bits needed for representing $\vec{n}$. For a weight function W and a weight η, define $N(W, \eta) = \max\{N(\vec{n}) : W(\vec{n}) \geq \eta\}$, and $\tilde{N}(W, \eta) = \max\left\{\tilde{N}(\vec{n}) : W(\vec{n}) \geq \eta)\right\}$. In [20], it was shown that the BFB decision variant can be solved using $O(\tilde{N}(\vec{n}))$ bit operations (i.e. linear time in the input length), the string search variant can be solved in $O(N(\vec{n}))$ operations (i.e. linear time in the output length), and that the vector search variant can be solved using at most a sub-exponential number of operations $2^{O(\log^2 N(W,\eta))}$. Here, we give algorithms for the two new exhaustive search variants. While theoretically the output of these algorithms can be exponential with respect to $N(W, \eta)$, we show that for realistic inputs this output is manageable. In addition, we describe an *Informed Search* (IS) approach that significantly reduces the running time in practice by eliminating irrelevant search paths and traversing only paths which are guaranteed to produce valid solutions. Next, we describe some ideas taken from [20], upon which the algorithms presented here are built.

An *l-BFB palindrome* is an l-BFB string of the form $\beta = \alpha\bar{\alpha}$. It can be shown that $\beta = \alpha\bar{\alpha}$ is an l-BFB palindrome if and only if α is an l-BFB string. By definition, $\varepsilon = \varepsilon\bar{\varepsilon}$ is an l-BFB palindrome for every $l \geq 1$. In addition, observe that when $\beta = \alpha\bar{\alpha}$ we have that $\vec{n}(\beta) = 2\vec{n}(\alpha)$. This allows to replace the question "is there a BFB string admitting the count vector $\vec{n}$" by the equivalent question "is there a BFB palindrome admitting the count vector $2\vec{n}$".

An *l-block* is a string of the form $\beta = \sigma_l\beta'\bar{\sigma}_l$, where β' is an $(l+1)$-BFB palindrome. It can be shown that an l-block is a special form of an l-BFB palindrome, and that every l-BFB palindrome is some palindromic concatenation of

l-blocks (though not every palindromic concatenation of l-blocks is a valid l-BFB palindrome). These observations allow to adopt a "layered" view of BFB palindromes, as follows (Fig. 3). Let $\beta = \alpha\overline{\alpha}$ be a 1-BFB palindrome, where $\vec{n}(\beta) = 2\vec{n}(\alpha) = [2n_1, 2n_2, \ldots, 2n_k]$. Therefore, β is a palindromic concatenation of 1-blocks, and denote by B^1 the collection of all these blocks. Every 1-block in B^1 is a string of the form $\mathrm{A}\beta'\bar{\mathrm{A}}$, where β' is some 2-BFB palindrome. As there are $2n_1$ occurrences of A and $\bar{\mathrm{A}}$ in β, and each block in B^1 contains exactly two such occurrences, the total number of blocks in B^1 is exactly n_1. Masking the letters A and $\bar{\mathrm{A}}$ from all blocks in B^1, the collection becomes a 2-BFB palindrome collection of size n_1. The 2-BFB palindromes in this collection can be further decomposed into 2-blocks, yielding a collection B^2 of 2-blocks. Similarly as above, B^2 contains exactly n_2 blocks. This process can continue inductively, yielding for every $1 \leq l \leq k$ a corresponding collection B^l of l-blocks, whose size is n_l. One may also imagine an additional collection in this series B^{k+1}, containing zero $(k+1)$-blocks.



(a) A BFB process (b) Palindrome layers

Fig. 3. (a) A BFB process generating a string α: $\mathrm{ABCD} \xrightarrow{\mathrm{BFB}} \mathrm{ABCD}\mathbf{\bar{D}} \xrightarrow{\mathrm{BFB}} \mathrm{ABCD\bar{D}}\mathbf{D\bar{D}\bar{C}\bar{B}} \xrightarrow{\mathrm{BFB}} \mathrm{ABCD\bar{D}D\bar{D}\bar{C}\bar{B}}\mathbf{B} \xrightarrow{\mathrm{BFB}} \mathrm{ABCD\bar{D}D\bar{D}\bar{C}\bar{B}B}\mathbf{\bar{B}BC}$. (b) The layers of the BFB palindrome $\beta = \alpha\overline{\alpha}$. The blocks in each layer are marked with annotations of the form β_i.

This layered view is exploited in a reversed order by the algorithms in [20], developing a BFB palindrome given an input count vector $\vec{n} = [n_1, n_2, \ldots, n_k]$: Starting with an empty collection B^{k+1} of $(k+1)$-blocks, the algorithm computes iteratively a sequence of collections $B^k, B^{k-1}, \ldots, B^1$, each collection B^l is an l-block collection of size n_l. In order to generate B^l, the algorithm first concatenates $(l+1)$-blocks from B^{l+1}, forming a collection B of $(l+1)$-BFB palindromes of size n_l (this procedure is called *folding*). Then, each $(l+1)$-BFB palindrome $\beta' \in B$ is wrapped with a pair of σ_l segments, rendering it into an l-block $\beta = \sigma_l\beta'\overline{\sigma}_l$, and B^l is set to be the collection containing all these l-blocks. The final collection of 1-blocks B^1 is folded one more time into a single 1-BFB palindrome $\beta = \alpha\overline{\alpha}$, and the algorithm returns the half-length prefix α of this palindrome as a BFB string admitting the input count vector $\vec{n}$.

Fig. 3b illustrates a possible run of the algorithm over the input count vector $\vec{n} = [1, 5, 3, 4]$. First, the algorithm initializes an empty collection of 5-blocks B^5. In the first iteration, there is a need to perform concatenations of blocks in B^5, and produce $n_4 = 4$ 5-BFB palindromes. Such 5-BFB palindromes may only be obtained by concatenating zero elements (as there are no elements in B^5), and so four empty strings are generated in this folding process, yielding the 5-BFB palindrome collection $\{4\varepsilon\}$. Next, each 5-BFB palindrome in this collection is wrapped by $\sigma_4 = \mathrm{D}$ and $\overline{\sigma}_4 = \bar{\mathrm{D}}$, producing the collection of 4-blocks $B^4 = \left\{4\mathrm{D}\varepsilon\bar{\mathrm{D}}\right\} = \{4\beta_1\}$. In the next iteration, the collection B^4 needs to reduce its size from $n_4 = 4$ into $n_3 = 3$ by concatenating its elements to produce 3-BFB palindromes. In this example, there are two concatenations of two elements the form $\beta_1\beta_1$, and one concatenation of zero elements that produces an empty string ε. The 4-BFB palindromes in the resulting folded collection $\{2\beta_1\beta_1, \varepsilon\}$ are wrapped by $\sigma_3 = \mathrm{C}$ and $\overline{\sigma}_3 = \bar{\mathrm{C}}$, yielding the 3-block collection $B^3 = \left\{2\mathrm{C}\beta_1\beta_1\bar{\mathrm{C}}, \mathrm{C}\varepsilon\bar{\mathrm{C}}\right\} = \{2\beta_2, \beta_3\}$. This process continues for two more iterations, generating similarly the collections $B^2 = \{2\beta_4, \beta_5, 2\beta_6\}$ and $B^1 = \{\beta_7\}$. All elements in the last collection B^1 are then concatenated into a single 1-BFB palindrome β (in this example B^1 contains a single element β_7, and so $\beta = \beta_7$), and the returned string α is the half-length prefix of this palindrome.

The ability of the schematic algorithm above to process the entire input vector $\vec{n}$ and produce a corresponding BFB string depends on its ability to fold intermediate collections B^l computed along its run. In cases where it cannot fold some intermediate block collection, it returns a fail message, implying no BFB string admits the input vector $\vec{n}$. A case where folding cannot be applied is for example the case where $n_2 = 2$, $B^2 = \{\mathrm{BC}\bar{\mathrm{C}}\bar{\mathrm{B}}, \mathrm{B}\bar{\mathrm{B}}\}$, and $n_1 = 1$. In this case, since both possible concatenations $\mathrm{BC}\bar{\mathrm{C}}\bar{\mathrm{B}}\mathbf{B}\bar{\mathbf{B}}$ and $\mathbf{B}\bar{\mathbf{B}}\mathrm{BC}\bar{\mathrm{C}}\bar{\mathrm{B}}$ of the two elements in B^2 are non-palindromic, the folding procedure must fail at this stage. Another example of a fail folding is the case where $n_2 = 3$, $B^2 = \{\mathrm{BC}\bar{\mathrm{C}}\bar{\mathrm{B}}, 2\mathrm{B}\bar{\mathrm{B}}\}$, and $n_1 = 1$. In this case, though there exists a palindromic concatenation $\mathbf{B}\bar{\mathbf{B}}\mathrm{BC}\bar{\mathrm{C}}\bar{\mathrm{B}}\mathbf{B}\bar{\mathbf{B}}$ of all three elements in B^2, this concatenation is not a valid BFB palindrome (since any 2-BFB string containing the segment C must start with the prefix BC), and so the collection may not be folded.

In [20], it was shown that the ability to fold a block collection depends on a property called the *signature* of the collection. A signature of an l-BFB palindrome collection is an infinite sequence of integers $\vec{s} = [s_0, s_1, s_2, \ldots]$ with the following properties: (1) the first nonzero element in $\vec{s}$ (if there is such an element) must be positive, (2) the *cardinality* of the signature, defined by $\|\vec{s}\| = \sum_{d=0}^{\infty} 2^d \mathrm{abs}(s_d)$ (where $\mathrm{abs}(s_d)$ is the absolute value of s_d), equals to the size of the collection to which the signature corresponds, and (3) wrapping the collection (i.e. replacing each l-BFB palindrome β in the collection with an $(l-1)$-block $\sigma_{l-1}\beta\overline{\sigma}_{l-1}$) does not change its signature. In this sense, a signature can be thought of as a generalization of a binary representation of an integer, in which the coefficients may be other integers besides 0 and 1 (with the additional restriction of a positive first nonzero element, and the fact the absolute coefficient value is taken when computing the corresponding summation). The prefix of a signature $\vec{s}$ up to its d-th element is denoted by $\vec{s}_d = [s_0, s_1, \ldots, s_d]$. Due to being relatively technical, we omit here the formal signature definition and refer intrigue readers to [20] for a full explanation on how to derive collection signatures.

From the signature cardinality definition, it follows that for a signature $\vec{s}$ such that $\|\vec{s}\| = n$, all signature elements s_i for $i > \log n$ are zeros, thus signatures can be explicitly represented by a (small) finite number of nonzero elements. In addition, it follows that the only signature of an empty collection is $\vec{s} = [0, 0, \ldots]$, and the only signature of a collection containing a single element is $\vec{s} = [1, 0, \ldots]$ (the "…" notation implies that the remaining signature elements are zeros). Otherwise, two collections of the same size may have different signatures. Signatures can be ranked according to their *lexicographic order*. That is, say that $\vec{s} < \vec{s}'$ if there exists an index d such that $\vec{s}_{d-1} = \vec{s}'_{d-1}$ and $s_d < s'_d$, and say that $\vec{s} \leq \vec{s}'$ if $\vec{s} < \vec{s}'$ or $\vec{s} = \vec{s}'$.

Lemma 1. *Let B be an l-block collection with a signature $\vec{s}$. For any folding B' of B and its corresponding signature $\vec{s}'$, $\vec{s} \leq \vec{s}'$. In addition, for any signature $\vec{s}'$ such that (1) $\vec{s} \leq \vec{s}'$ and (2) $\vec{s}'$ is the lexicographically minimal signature among all signatures of cardinality $\|\vec{s}'\|$ that meet (1), there exists a folding B' of B whose signature is $\vec{s}'$.*

The proof of Lemma 1 follows from Claims 14 and 28 in [20] (Supporting Information). The signatures corresponding to the 4 block collections implied by the BFB palindrome presented in Fig. 3b are $\vec{s}^4 = [0, 0, 1, 0, \ldots]$, $\vec{s}^3 = [1, -1, 0, \ldots]$, $\vec{s}^2 = [1, 0, -1, 0, \ldots]$, and $\vec{s}^1 = [1, 0, \ldots]$, respectively. Observe that the cardinality of each signature equals to the size of the corresponding collection (or the corresponding count in $\vec{n} = [1, 5, 3, 4]$), and that $\vec{s}^{l+1} \leq \vec{s}^l$ for every $1 \leq l < 4$.

It follows from Lemma 1 that a vector $\vec{n} = [n_1, n_2, \ldots, n_k]$ is a BFB count vector if and only if there exists a series of lexicographically non-increasing signatures $\vec{s}^1, \vec{s}^2, \ldots, \vec{s}^k$ such that $n_l = \|\vec{s}^l\|$ for every $1 \leq l \leq k$, and the first signature in this series satisfies $\vec{s}^1 \leq [1, 0, \ldots]$ (the signature of a collection with one element, due to the last concatenation of all 1-blocks in B^1 into a single palindrome). Call such a signature series a *valid signature series* for $\vec{n}$, and so we get the following conclusion:

Conclusion 1. *A vector $\vec{n}$ is a BFB vector if and only if it has a valid signature series. Moreover, any sub-sequence of a BFB vector is also a BFB vector, evident by the corresponding sub-series of a valid signature series for the full vector.*

For example, the vector $\vec{n} = [3, 4]$ is a BFB vector, due to the valid signature series $\vec{s}^1 = [1, -1, 0, \ldots], \vec{s}^2 = [0, 0, 1, 0, \ldots]$. A corresponding BFB string may be obtained by AB $\xrightarrow{\text{BFB}}$ AB$\mathbf{\bar{B}}$ $\xrightarrow{\text{BFB}}$ AB$\bar{\text{B}}$$\mathbf{B\bar{B}\bar{A}}$ $\xrightarrow{\text{BFB}}$ AB$\bar{\text{B}}$B$\bar{\text{B}}\bar{\text{A}}$$\mathbf{A}$. An example for a vector that does not have a valid signature series is the vector

$\vec{n} = [4, 3]$: the only signatures with cardinality 4 are the signatures $[0, 0, 1, 0, \ldots]$, $[0, 2, 0, \ldots]$, $[2, 1, 0, \ldots]$, $[2, -1, 0, \ldots]$, and $[4, 0, \ldots]$. Among these signatures, the only ones who lexicographically precede the signature $[1, 0, \ldots]$ are the signatures $[0, 0, 1, 0, \ldots]$ and $[0, 2, 0, \ldots]$. Nevertheless, the only signatures of cardinality 3 are $[1, -1, 0, \ldots]$, $[1, 1, 0, \ldots]$, and $[3, 0, \ldots]$, and none of them precedes the two possible 4-cardinality signatures.

As a matter of fact, restating Algorithm DECISION-BFB in [20] (Supporting Information), one can describe it as follows. Setting $\vec{s}^{k+1}$ to be the the signature $[0, 0, \ldots]$ of an empty collection (which is also the lexicographically minimal among all signatures), the algorithm produces iteratively the signatures $\vec{s}^k, \ldots, \vec{s}^1$ in a valid signature series for the input vector $\vec{n} = [n_1, n_2, \ldots, n_k]$. Each signature $\vec{s}^l$ is obtained by applying the minimal lexicographic increment to $\vec{s}^{l+1}$ so that it would admit the cardinality $\|\vec{s}^l\| = n_l$. The algorithm returns true if and only if all increments are successful.

3 Algorithms

In this section we develop algorithms for the two exhaustive search variants of the BFB problem. To do so, we first describe some ideas and subroutines that would allow efficient implementations of these algorithms.

Let $\vec{n} = [n_1, n_2, \ldots, n_k]$ be a BFB vector, and let $1 \leq l \leq k + 1$. Define the *right-maximal signature* $R(\vec{n}_{1,l-1})$ of the prefix $\vec{n}_{1,l-1} = [n_1, n_2, \ldots, n_{l-1}]$ of $\vec{n}$ to be $[1, 0, \ldots]$ if $l = 1$, and otherwise to be the lexicographically maximal signature $\vec{s}^{l-1}$ in some valid signature series $\vec{s}^1, \ldots, \vec{s}^{l-1}$ for $\vec{n}_{1,l-1}$. Similarly, define the *left-minimal signature* $L(\vec{n}_{l,k})$ of the suffix $\vec{n}_{l,k} = [n_l, n_{l+1}, \ldots, n_k]$ of $\vec{n}$ to be $[0, 0, \ldots]$ if $l = k+1$, and otherwise to be the lexicographically minimal signature $\vec{s}^l$ in some valid signature series $\vec{s}^l, \ldots, \vec{s}^k$ for $\vec{n}_{l,k}$.

Lemma 2. *Let $\vec{n} = [n_1, n_2, \ldots, n_k]$ be a BFB vector. For every $1 \leq l' \leq l \leq k + 1$, $L(\vec{n}_{l,k}) \leq R(\vec{n}_{1,l-1})$, $R(\vec{n}_{1,l-1}) \leq R(\vec{n}_{1,l'-1})$, and $L(\vec{n}_{l,k}) \leq L(\vec{n}_{l',k})$.*

Proof. We start by showing the first inequality in the lemma. If $l = 1$ or $l = k+1$, $L(\vec{n}_{l,k}) \leq R(\vec{n}_{1,l-1})$ follows immediately. Otherwise, consider a valid signature series $\vec{s}^1, \vec{s}^2, \ldots, \vec{s}^k$ for $\vec{n}$. Note that its prefix $\vec{s}^1, \vec{s}^2, \ldots, \vec{s}^{l-1}$ is a valid signature series for $\vec{n}_{1,l-1}$, and its suffix $\vec{s}^l, \vec{s}^{l+1}, \ldots, \vec{s}^k$ is a valid signature series for $\vec{n}_{l,k}$. Thus, by definition, $L(\vec{n}_{l,k}) \leq \vec{s}^l \leq \vec{s}^{l-1} \leq R(\vec{n}_{1,l-1})$.

To show the second inequality in the lemma, let $\vec{s}^1, \vec{s}^2, \ldots, \vec{s}^{l-1}$ be a valid signature series for $\vec{n}_{1,l-1}$ such that $\vec{s}^{l-1} = R(\vec{n}_{1,l-1})$. Observe similarly as above that $R(\vec{n}_{1,l-1}) = \vec{s}^{l-1} \leq \vec{s}^{l'-1} \leq R(\vec{n}_{1,l'-1})$. The last inequality in the lemma is shown symmetrically. □

The MIN-DECREMENT procedure (Algorithm 1) gets as an input a signature $\vec{s}$ and an integer $n \geq 0$, and returns the lexicographically maximal signature $\vec{s}'$ such that $\vec{s}' \leq \vec{s}$ and $\|\vec{s}'\| = n$ if such a signature exists, and otherwise returns a fail message. Here, for an integer $m \neq 0$, the notation d_m represents the *parity degree* of m, which is defined to be the maximum integer d_m such that m divides by 2^{d_m}. Thus, for example, $d_{13} = d_{13 \cdot 2^0} = 0$, and $d_{-12} = d_{-3 \cdot 2^2} = 2$. The correctness of this computation is shown in the online Appendix. Symmetrically, the MIN-INCREMENT procedure gets as an input a signature $\vec{s}$ and an integer $n \geq 0$, and returns the lexicographically minimal signature $\vec{s}'$ such that $\vec{s} \leq \vec{s}'$ and $\|\vec{s}'\| = n$ if such a signature exists, and otherwise returns a fail message. The pseudo-code for this procedure is given in the online Appendix, and its proof is symmetric to that of the MIN-DECREMENT procedure.

Algorithm 1. MIN-DECREMENT($\vec{s}, n$)

```
Input: A signature $\vec{s}$ and an integer $n \geq 0$.
Output: The lexicographically maximal signature $\vec{s}' \leq \vec{s}$ such that $\|\vec{s}'\| = n$, or the
        message "FAILED" if there is no such signature.
1 Let $m = \|\vec{s}\| - n$. If $m = 0$ then return $\vec{s}$.
2 Else if there is an integer $0 \leq d \leq d_m$ such that $n \geq \|\vec{s}_{d-1}\| + 2^d \max\{-s_d + 1, 0\}$ then
3     Let $d$ be the maximum integer meeting the condition above. Initialize $\vec{s}'$ so that
      $\vec{s}'_{d-1} = \vec{s}_{d-1}$, and $s'_d = s_d - 2$ if $d < d_m$, or $s'_d = s_d - 1$ if $d = d_m$.
4     If $n \geq \|\vec{s}'_d\|$ then set $s'_{d+1} \leftarrow \frac{n - \|\vec{s}'_d\|}{2^{d+1}}$.
5     Else set $s'_d \leftarrow \frac{n - \|\vec{s}'_{d-1}\|}{2^d}$.
6     Return $\vec{s}'$.
7 Else return "FAILED".
```

Lemma 3. *If $\vec{n}_{1,l-1} = [n_1, \ldots, n_{l-1}]$ is a BFB vector, $\vec{s} = R(\vec{n}_{1,l-1})$, and $\vec{s}' =$ MIN-DECREMENT$(\vec{s}, n_l)$, then $\vec{s}'$ is the right-maximal signature for the BFB vector $\vec{n}_{1,l} = [n_1, \ldots, n_{l-1}, n_l]$. Symmetrically, if $\vec{n}_{l+1,k} = [n_{l+1}, \ldots, n_k]$ is a BFB vector, $\vec{s} = L(\vec{n}_{l+1,k})$, and $\vec{s}' =$ MIN-INCREMENT$(\vec{s}, n_l)$, then $\vec{s}'$ is the left-minimal signature for the BFB vector $\vec{n}_{l,k} = [n_l, n_{l+1}, \ldots, n_k]$.*

Proof. We show the first part of the lemma, where the second part is shown symmetrically. First, note that the constructed vector $\vec{n}_{1,l}$ is indeed a BFB vector, due to the corresponding valid signature series obtained by adding $\vec{s}'$ to a valid signature series for $\vec{n}_{l-1}$ whose last signature is $\vec{s}$. Note that $\|R(\vec{n}_{1,l})\| = \|\vec{s}'\| = n_l$. From Lemma 2 $R(\vec{n}_{1,l}) \leq \vec{s}$, and since $\vec{s}' =$ MIN-DECREMENT$(\vec{s}, n_l)$ it follows that $R(\vec{n}_{1,l}) \leq \vec{s}'$. From the maximality of $R(\vec{n}_{1,l})$, $R(\vec{n}_{1,l}) = \vec{s}'$. □

In the rest of this section, let W be a weight function, and $0 < \eta \leq 1$ some weight threshold. Let $0 \leq l \leq k$, and consider the set of all signature-weight pairs of the form $\langle R(\vec{n}_{1,l}), W(\vec{n}_{1,l})\rangle$ such that $\vec{n}_l = [n_1, n_2, \ldots, n_l]$ is a BFB vector and $W(\vec{n}_{1,l}) \geq \eta$. Say that the pair $\langle \vec{s}, w\rangle$ within this set *dominates* the pair $\langle \vec{s}', w'\rangle$ if $\vec{s}' \leq \vec{s}$ and $w' \leq w$. Define the *l-th boundary curve* C^l with respect to W and η as the maximal subset of these pairs satisfying that no pair in C^l dominates another pair in C^l, and note that C^l is unique. Traversing the pairs in C^l from lowest to highest lexicographic signature rank, the series of signature values strictly increases, while the series of weight values strictly decreases, yielding a steps-like curve (Fig. 4). Algorithm 2 generates all boundary curves for W and η, which will later be exploited by algorithms for the BFB exhaustive vector and string search variants.

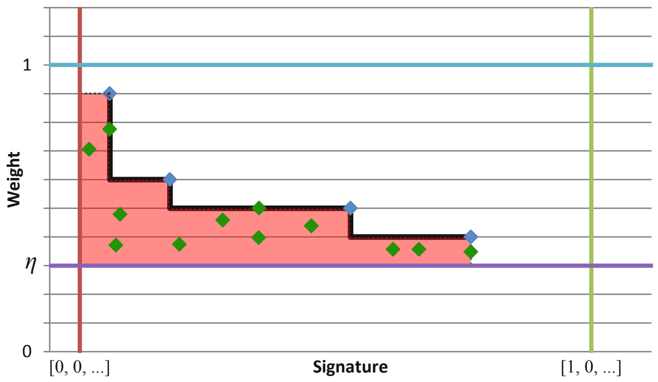


Fig. 4. A boundary curve. Points correspond to pairs of the form $\langle \vec{s}, w\rangle$, with x-coordinate reflecting the lexicographic rank of $\vec{s}$ and y-coordinate equals to w. Blue points belong to the boundary curve, and green points are dominated by points on the curve.

Algorithm 2. BOUNDARY-CURVES (W, η)

```
Input: A weight function $W$ and a weight $\eta$.
Output: All boundary curves for $W$ and $\eta$.
1 Set $C^0 \leftarrow \{\langle [1,0,\ldots], 1\rangle\}$.
2 For $l \leftarrow 1$ to $k$ do
3     Set $C^l \leftarrow \emptyset$.
4     For each $n$ and $\langle \vec{s}', w'\rangle \in C^{l-1}$ s.t. $w' \cdot w_{l,n} \geq \eta$ and MIN-DECREMENT($\vec{s}', n$) does not fail do
5         Let $\vec{s}$ be the output of MIN-DECREMENT($\vec{s}', n$), and let $w = w' \cdot w_{l,n}$.
6         If $\langle \vec{s}, w\rangle$ is not dominated by any pair in $C^l$ then
7             Add $\langle \vec{s}, w\rangle$ into $C^l$, and remove from $C^l$ all pairs dominated by $\langle \vec{s}, w\rangle$.
8 Return $\left\{C^0, C^1, \ldots, C^k\right\}$.
```

Proof (Algorithm 2). Note that a pair in C^0 corresponds to a right-maximal signature and a weight of an empty vector. By definition, the only such pair is the pair $\langle [1,0,\ldots], 1\rangle$, and the algorithm correctly sets C^0 to contain this single pair (line 1). Now, assuming inductively the algorithm has computed correctly the curve C^{l-1}, we prove it also computes correctly C^l. It is clear from lines 6 and 7 of the algorithm that no pair in the set C^l computed by the algorithm dominates another pair in this set. It is therefore remains to show that after the l-th loop iteration was executed (1) for every BFB vector $\vec{n}_{1,l} = [n_1, \ldots, n_l]$ with $W(\vec{n}_{1,l}) \geq \eta$ there exists a pair $\langle \vec{s}, w\rangle \in C^l$ which dominates $\langle R(\vec{n}_{1,l}), W(\vec{n}_{1,l})\rangle$, and (2) for every pair $\langle \vec{s}, w\rangle \in C^l$ there exists some BFB vector $\vec{n}_{1,l} = [n_1, \ldots, n_l]$ such that $\vec{s} = R(\vec{n}_{1,l})$ and $w = W(\vec{n}_{1,l})$.

We start by showing (1). Let $\vec{n}_{1,l} = [n_1, \ldots, n_{l-1}, n_l]$ be a BFB vector with $W(\vec{n}_{1,l}) \geq \eta$, and consider its prefix $\vec{n}_{1,l-1} = [n_1, \ldots, n_{l-1}]$. Observe that $W(\vec{n}_{n,l-1}) = \frac{W(\vec{n}_{1,l-1})}{w_{l,n_l}} \geq \eta$. As $\vec{n}_{1,l-1}$ is also a BFB vector, the inductive assumption implies that C^{l-1} contains a pair $\langle \vec{s}', w' \rangle$ that dominates $\langle R(\vec{n}_{1,l-1}), W(\vec{n}_{1,l-1}) \rangle$. From Lemma 2, $R(\vec{n}_{1,l}) \leq R(\vec{n}_{1,l-1}) \leq \vec{s}'$. Since $\|R(\vec{n}_{1,l})\| = n_l$, running MIN-DECREMENT $(\vec{s}', n_l)$ does not fail, and returns a signature $\vec{s}$ such that $R(\vec{n}_{1,l}) \leq \vec{s} \leq \vec{s}'$ and $\|\vec{s}\| = n_l$. As $w' \cdot w_{l,n_l} \geq W(\vec{n}_{1,l-1}) \cdot w_{l,n_l} = W(\vec{n}_{1,l}) \geq \eta$, it follows that the algorithm runs the code in lines 5-7 with respect to n_l and $\langle \vec{s}', w' \rangle$. In particular, the algorithm updates C^l with the pair $\langle \vec{s}, w \rangle$ for $w = w' \cdot w_{l,n_l} \geq W(\vec{n}_{1,l})$ (lines 6-7). Therefore, at the end of the l-th iteration, either C^l contains $\langle \vec{s}, w \rangle$, or it contains some other signature-weight pair that dominates $\langle \vec{s}, w \rangle$, and so it contains a pair that dominates $\langle R(\vec{n}_{1,l}), W(\vec{n}_{1,l}) \rangle$.

To show (2), assume that C^l contains a pair $\langle \vec{s}, w \rangle$. This pair was added to C^l in line 7 of the algorithm, which means there exists some pair $\langle \vec{s}', w' \rangle \in C^{l-1}$ such that for $n_l = \|\vec{s}\|$, $\vec{s} =$ MIN-DECREMENT$(\vec{s}', n_l)$, and $w = w' \cdot w_{l,n_l} \geq \eta$. From the inductive assumption, there is BFB vector $\vec{n}_{1,l-1} = [n_1, \ldots, n_{l-1}]$ such that $\vec{s}' = R(\vec{n}_{1,l-1})$ and $w' = W(\vec{n}_{1,l-1})$. For the vector $\vec{n}_{1,l} = [n_1, \ldots, n_{l-1}, n_l]$, lemma 3 implies that $\vec{s} = R(\vec{n}_{1,l})$. In addition $W(\vec{n}_{1,l}) = w$, and the lemma follows. □

Finally, we present Algorithm 3 for the BFB exhaustive vector search variant. The algorithm processes the segments of the input one by one, starting from the k-th segment down to the first segment. The notation $[n, \vec{n}]$ is used for denoting a vector whose first element is the integer n, and its remaining suffix is the vector $\vec{n}$.

Algorithm 3. EXHAUSTIVE-VECTOR-SEARCH (W, η)

```
Input: A weight function $W$ and a weight $0 < \eta \leq 1$.
Output: All BFB vectors $\vec{n} = [n_1, n_2, \ldots, n_k]$ satisfying $W(\vec{n}) \geq \eta$.
1 Generate all boundary curves $C^0, C^1, \ldots, C^k$ with respect to $W$ and $\eta$ using Algorithm 2. If
  $C^k$ is empty, return the message "NO SOLUTION" and halt.
2 Set $Q^{k+1}$ to be the collection containing a single empty vector.
3 For $l \leftarrow k$ down to 1 do
4     Set $Q^l \leftarrow \emptyset$.
5     For each $\vec{n}_{l+1,k} \in Q^{l+1}$ and count $n$ such that $W([n, \vec{n}_{l+1,k}]) \geq \eta$ and
      MIN-INCREMENT $(L(\vec{n}_{l+1,k}), n)$ does not fail do
6         Let $\vec{n}_{l,k} = [n, \vec{n}_{l+1,k}]$, and let $\vec{s} =$ MIN-INCREMENT$(L(\vec{n}_{l+1,k}), n)$.
7         If there exists a pair $\langle \vec{s}', w' \rangle \in C^{l-1}$ such that $\vec{s} \leq \vec{s}'$ and $w' \cdot W(\vec{n}_{l,k}) \geq \eta$ then
8             Add $\vec{n}_{l,k}$ to $Q^l$.
9 Return $Q^1$.
```

Proof (Algorithm 3). By definition, if the boundary curve C^k is empty, it implies there is no BFB vector $\vec{n} = [n_1, \ldots, n_k]$ with $W(\vec{n}) \geq \eta$. In this case, the algorithm correctly reports there is no solution to the input (line 1).

Otherwise, we show for every $1 \leq l \leq k+1$ that the following invariant holds: *After Q^l is fully computed, Q^l contains $\vec{n}_{l,k} = [n_l, \ldots, n_k]$ if and only if $\vec{n}_{l,k}$ is a suffix of some BFB vector $\vec{n} = [n_1, \ldots, n_k]$ of weight $W(\vec{n}) \geq \eta$.* In particular, this invariant proves that the returned value Q^1 (line 9) is indeed the solution for the BFB exhaustive vector search variant, and so it only remains to establish the correctness of the invariant.

For $l = k+1$, the fact that Q^{k+1} contains a single empty suffix (line 2) derives the invariant in a straightforward manner. Otherwise, assuming inductively the invariant holds with respect to Q^{l+1}, we prove it also holds with respect to Q^l.

Let $\vec{n} = [n_1, \ldots, n_k]$ be a BFB vector of weight $W(\vec{n}) \geq \eta$, and consider its two suffixes $\vec{n}_{l,k} = [n_l, n_{l+1} \ldots, n_k]$ and $\vec{n}_{l+1,k} = [n_{l+1} \ldots, n_k]$. From the inductive assumption, $\vec{n}_{l+1,k} \in Q^{l+1}$. From Lemma 3, $\vec{s} = L(\vec{n}_{l,k})$ satisfies that $\vec{s} =$ MIN-INCREMENT$(L(\vec{n}_{l+1,k}), n_l)$. Since $W(\vec{n}_{l,k}) \geq W(\vec{n}) \geq \eta$, the condition in line 5 holds, and lines 6-8 are executed with respect to $\vec{n}_{l,k}$ and $\vec{s}$. Note that the prefix $\vec{n}_{1,l-1} = [n_1, \ldots, n_{l-1}]$ of $\vec{n}$ is a BFB vector with $W(\vec{n}_{1,l-1}) \geq W(\vec{n}) \geq \eta$. From the definition of C^{l-1}, there exists a pair $\langle \vec{s}', w' \rangle \in C^{l-1}$ that dominates the pair $\langle R(\vec{n}_{1,l-1}), W(\vec{n}_{1,l-1}) \rangle$. From Lemma 2, $L(\vec{n}_{l,k}) \leq R(\vec{n}_{1,l-1}) \leq \vec{s}'$. In addition, $w' \cdot W(\vec{n}_{l,k}) \geq W(\vec{n}_{1,l-1}) \cdot W(\vec{n}_{l,k}) = W(\vec{n}) \geq \eta$, and so the condition in line 7 holds, and the algorithm adds $\vec{n}_{l,k}$ into Q^l in line 8.

For the other direction of the invariant, let $\vec{n}_{l,k} = [n_l, n_{l+1}, \ldots, n_k] \in Q^l$. Due to the manner it was constructed (lines 5-6), its suffix $\vec{n}_{l+1,k} = [n_{l+1}, \ldots, n_k]$ is in Q^{l+1}, and from Lemma 3, $\vec{n}_{l,k}$ is a BFB vector with $L(\vec{n}_{l,k}) = \vec{s}$. From line 7, there exists a pair $\langle \vec{s}', w' \rangle \in C^{l-1}$ such that $\vec{s} \leq \vec{s}'$ and $w' \cdot W(\vec{n}_{l,k}) \geq \eta$, and so from the definition of C^{l-1} there exists a BFB vector $\vec{n}_{1,l-1} = [n_1, \ldots, n_{l-1}]$ for which $R(\vec{n}_{1,l-1}) = \vec{s}'$ and $W(\vec{n}_{1,l-1}) = w'$. The concatenation of $\vec{n}_{1,l-1}$ and $\vec{n}_{l,k}$ gives the vector $\vec{n} = [n_1, \ldots, n_{l-1}, n_l, \ldots, n_k]$, whose weight satisfies $W(\vec{n}) = W(\vec{n}_{1,l-1}) \cdot W(\vec{n}_{l,k}) = w' \cdot W(\vec{n}_{l,k}) \geq \eta$. In addition, $\vec{n}$ is a BFB vector, due to the corresponding valid signature series obtained by concatenating a valid signature series for $\vec{n}_{1,l-1}$ that ends with $\vec{s}'$ and a valid signature series for $\vec{n}_{l,k}$ that starts with $\vec{s}$, concluding this direction of the proof. $\square$

The algorithm for the exhaustive BFB string search variant applies a similar approach in order to produce all BFB strings whose count vector weights are at least η. It starts by generating signature curves exactly as done by Algorithm 3. Then, in each iteration l, instead of computing a set Q^l of count vectors, the algorithm computes a set P^l of l-block collections. At the end of the iteration, P^l contains all l-block collections B^l such that there exists some 1-BFB palindrome β in which the l-th layer's block collection is B^l, and the weight of the vector $\vec{n}$ such that $\vec{n}(\beta) = 2\vec{n}$ satisfies $W(\vec{n}) \geq \eta$. The initial collection P^{k+1} contains a single empty $(k+1)$-block collection. In the l-th iteration, for each $(l+1)$-block collection $B^{l+1} \in P^{l+1}$, all possible foldings of B^{l+1} are enumerated. For each such folding, its signature and weight are examined against C^{l-1} similarly as done in line 7 of Algorithm 3, and if meeting the condition all elements in the collection are wrapped, and the resulting l-block collection B^l is added into P^l. Due to space limits, we omit the details for the process of enumerating all foldings of B^{l+1}, which will be described in an extended version of this manuscript.

4 Results

In order to test our algorithms we have used cancer data taken from the Cancer Genome Project dataset [2]. This data covers aCGH samples (Affymetrix Genome-Wide Human SNP Array 6.0) from 746 human cancer cell lines. Segmentation and segment copy numbers are as reported by [2], who used the PICNIC software [8] for this analysis. In total, the dataset contains about 35 thousands chromosomal arms (746 samples, 23 or 24 chromosomes per sample, two arms per chromosome), each arm is segmented, and each segment is assigned an estimated copy number based on the observed aCGH data. As shown in [20], short BFB-like count vectors have a high probability to emerge even when the genome was rearranged with mechanisms different from BFB. Thus, in order to detect significant BFB evidence we have filtered the set of chromosomal arms to include only arms with at least eight consecutive segments such that no adjacent segments share the same copy number estimation. After this filtration, the remaining subset included 6589 chromosomal arms. As the estimated counts reflect the expected segment copy numbers in all copies of the chromosome in the sample, we have corrected the counts by reducing $p-1$ from each count, where p is the ploidy (i.e. the number of copies) of the chromosome in the sample. Typically $p = 2$, but since these are heavily rearranged cancer genomes, chromosomal losses and whole chromosomal duplications are not rare. We therefore allowed the value of p to vary between 1 and 5, and run the BFB analyses for each value.

As currently no analysis tool available produces count weights, we have derived such weights from the expected counts reported by PICNIC (after correcting for ploidy). Specifically, for a segment whose observed count is n, the weight of a count n' was defined by $\frac{\Pr(n|n')}{\Pr(n|n)}$, where $\Pr(x|\lambda) = \frac{\lambda^x e^{-\lambda}}{x!}$ is the probability to observe the value x for a random variable distributing according to the *Poisson* distribution with parameter λ. For each of the obtained weight functions, we used the DISTANCE-BFB algorithm from [20] to report all longest BFB sub-vectors with weight at least $\eta = 0.7$. Out of the 6589 samples, 54 samples had for at least one ploidy value $1 \leq p \leq 5$ a BFB sub-vector of length at least 8. Some samples had long BFB sub-vectors with respect to more than one ploidy value, and the total number of obtained BFB vectors was 86.

Then, we considered the segment coordinates and weight functions corresponding to the obtained sub-vectors, and run Algorithm 3 in order to find all BFB vectors of weights at least $\eta = 0.7$ with respect to these weight functions. For these 86 instances, a total number of 19154 heavy BFB vectors were found, with an average of 222 solutions per-instance. This reviles an interesting property of the problem when applied over this data: the vast majority of samples, 6535 out of 6589, cannot be explained by any BFB count vector (and thus are unlikely to be obtained from BFB), yet each one of those 54 samples who can be explained by BFB has about several tens or hundreds of corresponding count vectors.

The above analysis was run by two variants of our algorithm - the IS variant described by Algorithm 3, and a variant that runs a similar procedure without applying the IS optimization (essentially, it runs the same code as Algorithm 3, with the exception it does not generate the signature curves in line 1, and does not apply the condition in line 7 before adding new elements to collections Q^l). The disadvantage of the non-IS variant is in that sets of the form Q^l maintain BFB vectors $\vec{n}_{l,k} = [n_l, \ldots, n_k]$ which may not be suffixes of some BFB vectors $\vec{n} = [n_1, \ldots, n_k]$ of weight at least η. To measure the gain of the IS algorithm, we count the number of signature increment attempts the algorithms perform (line 5). On average, the IS variant performed 57-fold less increments, with a total number of 5672346 incrementation attempt over all 86 vectors, versus 325343441 for the non-IS algorithm. While the IS variant has a clear efficiency advantage over the non-IS variant, this advantage might be considered more modest than expected. A possible reason for that is that maximum copy number values reported in [2] were limited to 14, even when the data suggests higher copy numbers. In general, higher copy numbers usually imply a higher number of alternative heavy counts, which in turn induce a higher number of possible heavy count vectors. For example, when comparing the two algorithms over the synthetic count vector $\vec{n} = [3, 8, 111, 8, 5, 150, 11, 170, 4, 53, 100, 75, 49, 10, 42, 18]$, using the same Poisson-based weights as described above and requiring that output vectors weigh at least $\eta = 0.85$, the non-IS algorithm runs 218 second[1] and performs over 20 million signature increments, whereas the IS algorithm runs 120 milliseconds and performs 635 signature increments. Both algorithms return exactly the same output - a set of 18 BFB vectors. Other simulated inputs can cause memory explosion for the non-IS variant, while handled efficiently by the IS variant.

5 Discussion and Conclusions

The problem of detecting breakage fusion bridge is challenging, but significant progress has been made in the last few years. Our work suggests that while rare, BFB does occur in tumor derived cell lines and also in primary tumors. In this work, we describe algorithms that can be used to enumerate all possible BFB architectures given uncertain copy number data.

The results of our analyses heavily depend on the input weights, which in turn depend on separated analyses applied to biological data. While we used here a simple Poisson-based model in order to render fixed available count estimations into weight functions, it is clear that more realistic weighing can be applied. Examining Fig. 2 for example, one can observe that different segments demonstrate different variance in signal intensities, implying that some count estimates are more reliable than others. Incorporating segment lengths and signal variance information when choosing count weights is likely to produce more meaningful weights and improve the quality of the analyses output.

[1] Running time was measured for an intel Core i7 processor with Microsoft Windows 7 operating system, code is implemented in Java.

Different measurements can yield other types of BFB evidence. For example, deep sequencing experiments can sequence reads spanning genomic breakpoints. In a BFB modified genome, it is expected that many of these breakpoints reflect fold-back inversions (i.e. concatenations between reference segments and their inverted form), while such fold-back patterns are less common in other rearrangement mechanisms [4]. Thus, identification of high or low fold-back pattern frequencies can support or weaken the conjecture BFB has occurred, respectively. Such evidence is less frequent in currently available data, as reliable breakpoint information requires sequencing to a relatively high depth of coverage (while copy number data can be obtained also from sequencing with a lower depth of coverage or from aCGH experiments). When given though, such information can be integrated and improve the quality of BFB calling [20].

Acknowledgements. The authors are thankful to the anonymous RECOMB reviewers for their helpful comments. The research was supported by grants from the NIH (RO1-HG004962), and the NSF (CCF-1115206, IIS-1318386).

References

1. Alkan, C., Kidd, J.M., Marques-Bonet, T., Aksay, G., Antonacci, F., Hormozdiari, F., Kitzman, J.O., Baker, C., Malig, M., Mutlu, O., Sahinalp, S.C., Gibbs, R.A., Eichler, E.E.: Personalized copy number and segmental duplication maps using next-generation sequencing. Nat. Genet. 41, 1061–1067 (2009)
2. Bignell, G.R., Greenman, C.D., Davies, H., Butler, A.P., Edkins, S., Andrews, J.M., Buck, G., Chen, L., Beare, D., Latimer, C., Widaa, S., Hinton, J., Fahey, C., Fu, B., Swamy, S., Dalgliesh, G.L., Teh, B.T., Deloukas, P., Yang, F., Campbell, P.J., Futreal, P.A., Stratton, M.R.: Signatures of mutation and selection in the cancer genome. Nature 463(7283), 893–898 (2010)
3. Bignell, G.R., Santarius, T., Pole, J.C., Butler, A.P., Perry, J., Pleasance, E., Greenman, C., Menzies, A., Taylor, S., Edkins, S., Campbell, P., Quail, M., Plumb, B., Matthews, L., McLay, K., Edwards, P.A., Rogers, J., Wooster, R., Futreal, P.A., Stratton, M.R.: Architectures of somatic genomic rearrangement in human cancer amplicons at sequence-level resolution. Genome. Res. 17, 1296–1303 (2007)
4. Campbell, P.J., Yachida, S., Mudie, L.J., Stephens, P.J., Pleasance, E.D., Stebbings, L.A., Morsberger, L.A., Latimer, C., McLaren, S., Lin, M.L., McBride, D.J., Varela, I., Nik-Zainal, S.A., Leroy, C., Jia, M., Menzies, A., Butler, A.P., Teague, J.W., Griffin, C.A., Burton, J., Swerdlow, H., Quail, M.A., Stratton, M.R., Iacobuzio-Donahue, C., Futreal, P.A.: The patterns and dynamics of genomic instability in metastatic pancreatic cancer. Nature 467(7319), 1109–1113 (2010)
5. Carr, A.M., Paek, A.L., Weinert, T.: DNA replication: failures and inverted fusions. Semin. Cell Dev. Biol. 22(8), 866–874 (2011)
6. Chiang, D.Y., Getz, G., Jaffe, D.B., O'Kelly, M.J., Zhao, X., Carter, S.L., Russ, C., Nusbaum, C., Meyerson, M., Lander, E.S.: High-resolution mapping of copy-number alterations with massively parallel sequencing. Nat. Methods 6(1), 99–103 (2009)
7. Eckel-Passow, J.E., Atkinson, E.J., Maharjan, S., Kardia, S.L., de Andrade, M.: Software comparison for evaluating genomic copy number variation for Affymetrix 6.0 SNP array platform. BMC Bioinformatics 12, 220 (2011)

8. Greenman, C.D., Bignell, G., Butler, A., Edkins, S., Hinton, J., Beare, D., Swamy, S., Santarius, T., Chen, L., Widaa, S., Futreal, P.A., Stratton, M.R.: PICNIC: an algorithm to predict absolute allelic copy number variation with microarray cancer data. Biostatistics 11(1), 164–175 (2010)
9. Hanahan, D., Weinberg, R.A.: Hallmarks of cancer: the next generation. Cell 144(5), 646–674 (2011)
10. Hastings, P.J., Lupski, J.R., Rosenberg, S.M., Ira, G.: Mechanisms of change in gene copy number. Nat. Rev. Genet. 10(8), 551–564 (2009)
11. Kinsella, M., Bafna, V.: Combinatorics of the breakage-fusion-bridge mechanism. J. Comput. Biol. 19(6), 662–678 (2012)
12. Kitada, K., Yamasaki, T.: The complicated copy number alterations in chromosome 7 of a lung cancer cell line is explained by a model based on repeated breakage-fusion-bridge cycles. Cancer Genet. Cytogenet. 185, 11–19 (2008)
13. McClintock, B.: The Production of Homozygous Deficient Tissues with Mutant Characteristics by Means of the Aberrant Mitotic Behavior of Ring-Shaped Chromosomes. Genetics 23, 315–376 (1938)
14. McClintock, B.: The Stability of Broken Ends of Chromosomes in Zea Mays. Genetics 26, 234–282 (1941)
15. Medvedev, P., Stanciu, M., Brudno, M.: Computational methods for discovering structural variation with next-generation sequencing. Nat. Methods 6, 13–20 (2009)
16. Olshen, A.B., Venkatraman, E.S., Lucito, R., Wigler, M.: Circular binary segmentation for the analysis of array-based dna copy number data. Biostatistics 5(4), 557–572 (2004)
17. Reshmi, S.C., Roychoudhury, S., Yu, Z., Feingold, E., Potter, D., Saunders, W.S., Gollin, S.M.: Inverted duplication pattern in anaphase bridges confirms the breakage-fusion-bridge (bfb) cycle model for 11q13 amplification. Cytogenetic and Genome Research 116(1-2), 46–52 (2007)
18. Venkatraman, E.S., Olshen, A.B.: A faster circular binary segmentation algorithm for the analysis of array cgh data. Bioinformatics 23(6), 657–663 (2007)
19. Yoon, S., Xuan, Z., Makarov, V., Ye, K., Sebat, J.: Sensitive and accurate detection of copy number variants using read depth of coverage. Genome. Res. 19(9), 1586–1592 (2009)
20. Zakov, S., Kinsella, M., Bafna, V.: An algorithmic approach for breakage-fusion-bridge detection in tumor genomes. Proceedings of the National Academy of Sciences 110(14), 5546–5551 (2013)